HolidaiButler

Technical Architecture Documentation

Comprehensive system design for Mediterranean AI travel platform

Version 1.0 • Production-Ready Architecture • May 2025

System Architecture Overview

HolidaiButler employs a microservices architecture with cloud-native design principles, ensuring scalability, reliability, and optimal performance for Mediterranean travel experiences.

Multi-Layer Architecture

Presentation Layer

React Native App

iOS/Android Mobile

Progressive Web App

Responsive Web Interface

Admin Dashboard

Partner Management

API Gateway Layer

API Gateway

Request Routing

Authentication

JWT + OAuth 2.0

Rate Limiting

Traffic Management

Business Logic Layer

AI Service

Claude API Integration

Recommendation Engine

Personalization Logic

Booking Service

Reservation Management

Weather Service

Real-time Data

Data Layer

MongoDB Atlas

Primary Database

Redis Cache

Session & Cache

Elasticsearch

Search & Analytics

Infrastructure Layer

AWS/EU-West-1

Primary Cloud

Kubernetes

Container Orchestration

CloudFlare CDN

Global Delivery

⚙️ Technology Stack

🖥️ Frontend

- **React Native 0.72+** - Cross-platform mobile
- **TypeScript** - Type safety
- **Redux Toolkit** - State management
- **React Navigation 6** - Navigation
- **Styled Components** - UI styling

🔧 Backend

- **Node.js 18+** - Runtime environment
- **Express.js** - Web framework
- **TypeScript** - Development language
- **Prisma ORM** - Database abstraction
- **Socket.io** - Real-time communication

🗄️ Database

- **MongoDB Atlas** - Primary database
- **Redis** - Caching & sessions
- **Elasticsearch** - Search & analytics

- **PostgreSQL** - Financial data

🤖 AI & APIs

- **Claude 3.5 Sonnet** - Primary AI
- **OpenWeatherMap** - Weather data
- **Google Maps API** - Location services
- **Stripe API** - Payment processing

☁️ Infrastructure

- **AWS EU-West-1** - Primary cloud
- **Kubernetes (EKS)** - Container orchestration
- **Docker** - Containerization
- **Terraform** - Infrastructure as Code
- **CloudFlare** - CDN & security

🔒 Security

- **JWT Tokens** - Authentication
- **OAuth 2.0** - Social login
- **AWS KMS** - Key management
- **HTTPS/TLS 1.3** - Transport security
- **Helmet.js** - Security headers

API Architecture

RESTful API Design

Following OpenAPI 3.0 specification with comprehensive documentation and testing.

```
// Core API Endpoints Structure const apiRoutes = { // Authentication "POST /api/v1/auth/login": "User authentication", "POST /api/v1/auth/refresh": "Token refresh", "DELETE /api/v1/auth/logout": "User logout", // AI Chat "POST /api/v1/chat/message": "Send message to AI", "GET /api/v1/chat/history": "Retrieve chat history", "DELETE /api/v1/chat/clear": "Clear chat history", // Recommendations "GET /api/v1/recommendations": "Get personalized suggestions", "POST /api/v1/recommendations/feedback": "Rate recommendations", "GET /api/v1/pois/search": "Search points of interest", // Bookings "POST /api/v1/bookings/create": "Create new booking", "GET /api/v1/bookings/user": "Get user bookings", "PUT /api/v1/bookings/:id/cancel": "Cancel booking", // User Profile "GET /api/v1/user/profile": "Get user profile", "PUT /api/v1/user/profile": "Update user profile", "PUT /api/v1/user/preferences": "Update preferences" };
```

Real-time Communication

```
// WebSocket Events for Real-time Features const socketEvents = { // Chat Events "chat:message": "Real-time chat messages", "chat:typing": "Typing indicators", "chat:ai_response": "AI response streaming", // Location Events "location:update": "Location-based recommendations", "location:pois_nearby": "Nearby POIs update", // Booking Events "booking:confirmed": "Booking
```

```
confirmation", "booking:reminder": "Booking reminders",
// System Events "system:weather_update": "Weather
condition changes", "system:maintenance": "Maintenance
notifications" };
```

📊 Performance Specifications

<200ms

API Response Time

99.9%

Uptime SLA

10K+

Concurrent Users

<2s

AI Response Time

50GB

Monthly Data Transfer

15+

Supported Languages

Load Testing Results

```
// Performance Benchmarks (Simulated Load Testing)
const performanceResults = { concurrent_users: {
target: 1000, achieved: 1250, response_time_p95:
"180ms", error_rate: "0.1%" }, ai_processing: {
claude_api_latency: "1.2s average",
recommendation_generation: "800ms average",
cache_hit_ratio: "85%", fallback_activation: "<50ms"
}, database_performance: { read_operations: "50ms
p95", write_operations: "120ms p95", search_queries:
"200ms p95", connection_pool: "95% efficiency" },
infrastructure_metrics: { cpu_utilization: "65%
average", memory_usage: "4.2GB / 8GB",
```

```
    network_throughput: "150 Mbps peak",
    auto_scaling_trigger: "80% CPU for 5min" } };
```

🔒 Security Architecture

🛡️ Multi-Layer Security Implementation

Authentication & Authorization

- **JWT with RS256:** Secure token signing
- **OAuth 2.0 + PKCE:** Social login security
- **Refresh Token Rotation:** Token security
- **Multi-Factor Auth:** Optional 2FA
- **Session Management:** Secure logout

Data Protection

- **AES-256 Encryption:** Data at rest
- **TLS 1.3:** Data in transit
- **PII Tokenization:** Sensitive data
- **GDPR Compliance:** Privacy by design
- **Data Retention:** Automated purging

GDPR & Privacy Compliance

- **Data Processing Agreement:** Claude API compliance framework
- **Privacy by Design:** Built-in data protection
- **User Rights:** Access, rectification, erasure, portability
- **Consent Management:** Granular privacy controls
- **Audit Trail:** Complete data processing logs
- **Data Minimization:** Only necessary data collection

🚀 Deployment & DevOps

Development
Local testing & validation

↓

Staging
Integration testing

↓

QA
Quality assurance

↓

Production
Blue-green deployment

CI/CD Pipeline

```
# GitHub Actions Workflow name: Deploy HolidaiButler
on: push: branches: [main] jobs: test: runs-on:
ubuntu-latest steps: - uses: actions/checkout@v3 -
name: Run tests run: | npm ci npm run test:unit npm
run test:integration npm run test:e2e security: runs-
on: ubuntu-latest steps: - name: Security scan run: |
npm audit docker scan sonarqube-scan deploy: needs:
[test, security] runs-on: ubuntu-latest steps: - name:
Deploy to Kubernetes run: | kubectl apply -f k8s/
kubectl rollout status deployment/holidaibutler-api
```

📈 Monitoring & Analytics

🔍 Application Monitoring

- **Prometheus:** Metrics collection
- **Grafana:** Visualization dashboards
- **Jaeger:** Distributed tracing
- **ELK Stack:** Log aggregation
- **PagerDuty:** Alert management

📊 Business Analytics

- **User Behavior:** Journey tracking
- **AI Performance:** Response quality
- **Conversion Metrics:** Booking rates
- **Revenue Analytics:** Financial KPIs
- **Partner Performance:** Business metrics

📈 Scalability & Future Roadmap

🚀 Phase 1: Foundation (Months 1-6)

- **User Capacity:** 10K concurrent users
- **Geographic Coverage:** Costa Blanca region
- **AI Features:** Basic conversational travel planning
- **Integrations:** Core booking and weather services

⚡ Phase 2: Scale (Months 7-18)

- **User Capacity:** 100K concurrent users

- **Geographic Coverage:** Spanish Mediterranean coast
- **AI Features:** Advanced personalization, voice interface
- **Integrations:** Extended partner network, advanced analytics

🌍 Phase 3: European Expansion (Months 19-36)

- **User Capacity:** 500K concurrent users
- **Geographic Coverage:** Full Mediterranean region
- **AI Features:** Multi-cultural AI, predictive recommendations
- **Integrations:** International partnerships, enterprise solutions

Technical Documentation Support

For detailed implementation questions or additional technical specifications:

Architecture Team
tech@holidaibutler.com

Security Team
security@holidaibutler.com

DevOps Team
devops@holidaibutler.com
